

LQG controller design using GUI: Application to antennas and radio-telescopes

Erin Maneri^a and Wodek Gawronski^b

^a Montana State University, Bozeman, MT 59719, USA

^b Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA 91109

Abstract

The Linear Quadratic Gaussian (LQG) algorithm as presented in Refs.[1] and [2] has been used to control the JPL's beam wave-guide [2], and 70-m [3] antennas. This algorithm significantly improves tracking precision in a wind disturbed environment. Based on this algorithm and the implementation experience a Matlab based Graphical User Interface (GUI) was developed to design the LQG controllers applicable to antennas and radiotelescopes. The GUI is described in this paper. It consists of two parts: the basic LQG design and the fine-tuning of the basic design using a constrained optimization algorithm. The presented GUI was developed to simplify the design process, to make the design process user-friendly, and to enable design of an LQG controller for one with a limited control engineering background. The user is asked to manipulate the GUI sliders and radio buttons to watch the antenna performance. Simple rules are given at the GUI display.

1. Introduction

The NASA Deep Space Network (DSN) antennas serve as a communication tools for the space exploration. DSN antennas are located at three complexes: in California, Spain, and Australia. Viewed from outer space, the DSN complex looks like a cluster with one large (70-meter), and several smaller (34-meter) antennas. Antennas are the source of radio signals carrying commands and data to guide the actions of a spacecraft. As the Earth turns, the cluster begins to disappear over one edge, to be replaced by another at the opposite edge, which continues as the source of radio signals to the spacecraft. It is equally true that the same antennas are listening to any signal sent Earthward by a spacecraft. The very tiny amount of radio energy from the spacecraft is collected and focussed by the precision quasi-parabolic dish antennas into microwave equipment which amplifies it in low-noise amplifiers that operate at temperatures near absolute zero. From these amplifiers, the signal passes on to other equipment that eventually transforms it into a replica of the data that originated on the spacecraft. Thus, the DSN is a data communication service that accepts a stream of data to be transported to a spacecraft and delivers another stream of data that originated on the spacecraft.

An example of the NASA/JPL beam-wave guide (BWG) antenna with 34-meter dish is shown in Fig.1. The antenna can rotate with respect to the azimuth (vertical) and elevation (horizontal) axes. Rotation in azimuth is accomplished by moving the entire structure on a circular azimuth track

Precision pointing of the narrow signal to the spacecraft is critical, especially when making initial contact without having a received signal for reference. The required precision is proportional to the beamwidth of the signal, which is 22 millidegrees at S-band, 5.9 millidegrees at X-Band, and 1.5 millidegrees at Ka-Band. Thus, the high frequency (and most effective) Ka-band requires the most precise tracking.

Consider now radiotelescopes. A goal of today's radio astronomy is to detect radio waves emanating from gas and dust in the coldest regions of the universe - regions which emit insignificantly at wavelengths to which the human eye is sensitive ($1/2,000$ th of a millimeter), but which produce relatively strong signals at wavelengths near 1 millimeter. Large telescope size and ability to collect signals of very high frequency are the keys to achieve this goal. New radiotelescopes under construction: 100-meter Green Bank Telescope (GBT) in West Virginia and 50-meter Large Millimeter Wavelength Telescope (LMT) in Mexico will serve these purposes. The high-frequency signal requires a very high precision pointing, and large-size structure makes this goal rather difficult.

A controller that supervises the tracking of the antenna or radiotelescope shall guarantee the required tracking precision. This precision is achieved using an LQG (Linear Quadratic Gaussian) algorithm. The LQG controllers for antennas and radiotelescopes have been designed and implemented, see Refs. [1], [2], and [3]. Their design is an iterative process that requires experience, time, and patience (there is no explicit relationship between control design parameters and its performance). In order to make the process of the controller design simple and user-friendly a controller design

GUI was developed. This paper describes the Matlab based software, with two-stage controller design approach: basic LQG design and fine-tuning of the antenna LQG algorithm. The Matlab GUI presented here simplifies the design process, and creates a user-friendly environment in which one with a limited control engineering background may “play” to obtain a tracking algorithm of an excellent tracking performance.

2. Open-Loop Antenna Model

Prior to antenna controller design one must collect information pertaining to the open-loop antenna model. The antenna open loop (or rate-loop) model is obtained either from finite element analysis or from system identification procedures (that include field tests). The model consists of the antenna structure and the azimuth and elevation drives, as in Fig.2.

The inputs are azimuth and elevation rates, and wind gust disturbances. The outputs are azimuth and elevation encoder positions, and cross-elevation and elevation beam errors. The rate-loop model in this form is typically obtained from a finite element model. It is a two-axis model, in which wind gusts act directly on the structure surface.

The finite element model can be simplified by decoupling azimuth and elevation axes and consequently separating the azimuth and elevation models. This is justified by a weak cross coupling between azimuth command and elevation encoder, and between elevation command and azimuth encoder. Less than one one-thousandth of the coupling exists between azimuth and elevation command/encoder than does between command and encoder readings of either one of these axes alone.

The rate-loop models obtained from field tests are one-axis models. For these models cross-elevation and elevation beam position outputs are unavailable, and the wind disturbances are applied to the drive input rather than to the antenna structure. A typical model obtained from the field data is shown in Fig.3.

The rate-loop model is linear, and represented in the state-space form as the rate-loop triple (A, B, C) , where A is the state matrix, B is the input matrix, and C is the output matrix. Arbitrary coordinates can be used for the state space representation. For the convenience of the LQG controller design the GUI transforms the rate-loop triple into the modal coordinates. A description of the state space modal coordinates is given in Ref. [4].

3. Performance Criteria

The antenna performance is characterized mainly by its step responses, rate offset errors, wind disturbance rms errors, and transfer function bandwidth.

The step response is characterized by the settling time and the overshoot. The first of which is defined as the time at which the antenna encoder output remains within 3% threshold of the nominal value of the step command. A 14.8s settling time due to a unit step command is illustrated in Fig.4a. Overshoot (in percent) is the relative difference between the maximal encoder output and the commanded step with respect to the value of the commanded step. In Fig.4a the overshoot is 18%.

The magnitude and phase of the transfer function of the DSS26 antenna rate loop model (azimuth axis) is shown in Fig.5. One with a controls background can see that the model has one pole at zero (slope of the magnitude of the transfer function is -20 dB/dec for low frequencies). This pole is observed as a rigid-body rotation of the antenna with respect to azimuth axis, and in analytical terms, the system behaves as an integrator at low frequencies. The closed-loop transfer function of the antenna is shown in Fig.4b. Bandwidth is the frequency at which the magnitude drops 3 dB, or to 70.7% of its zero frequency level (which is 1). This is illustrated in Fig.4b, where bandwidth is 0.16 Hz.

The rate-offset response is characterized with the steady-state error. Fig.6 shows the 0.06 deg/s rate offset command and the antenna response. The steady-state error (lagging) is 0.012 deg.

Wind gusts action is characterized by the encoders rms (root-mean-square) errors.

Although we don't plot rms error separately, its value is computed in the simulation. For a visual display of an antenna encoder response to 20-mph wind gusts, see Fig.7 where there is an rms value of 0.10 mdeg.

4. Designing the LQG Controller

The controller design process consists of choosing the controller configuration and determining the controller gains.

4.1. Choosing the Controller Configuration

The LQG controller is a model-based controller. This means that the antenna rate-loop model is a part of the controller. It is used to estimate the non-measured antenna states using the measured rate input and the encoder output. In order to ensure estimation accuracy, the antenna model shall closely match the actual antenna dynamics (thus, the finite element model is unacceptable). Generally, the finite element model is used in the antenna design stage when a model of a "real" antenna is not yet available. For implementation purposes, the rate loop model obtained from the field test is used. This "real" model is obtained from an existing antenna by conducting field tests and system identification. The obtained model is given in the form of a state space representation.

The states of the LQG controller are divided into the tracking states (antenna position error and its integral), and the flexible mode states (flexible mode displacements). For this configuration the antenna state vector is of the following form:

$$x = \{e_i \quad e \quad x_a \quad x_{f11} \quad x_{f12} \quad \cdots \quad x_{fn1} \quad x_{fn2}\}^T$$

where e_i is the integral of the servo error, e is the servo error and x_a is a variable from the system identification model (of unexplained physical interpretation). Each flexible mode is described by two state variables, thus x_{f11} , x_{f12} are the two state components of the first natural mode, and x_{fn1} , x_{fn2} are the two state components of the n th natural mode. The BWG antennas typically have $n=5$ natural modes.

4.2. Determining LQG Controller Gains by Tuning its Weights

The LQG closed loop system block diagram is shown in Fig.8. It consists of an estimator that estimates the antenna states based on measured antenna input (u) and output (y). The estimator has its own gain, K_e , to ensure the minimal estimation error. The proportional (k_p), integral (k_i), and flexible mode (K_f) gains are also included in the model inputs to control the estimated state and the output. The design task is to determine the gains such that the antenna performance is optimal. The analytical background for the determination of the gains, and detailed block diagram of the LQG control system are given in the Appendix.

The process of determining the LQG gains is explained in detail in Ref.[1]. Here, we explain the approach in a heuristic way. The controller gains depend on the weight matrix Q and covariance matrix V . The weight matrix shapes the optimization index, a positive variable to be minimized. The covariance matrix specifies the noise in the system, and it impacts the estimator gains. The difficulty arises when one tries to express the performance of the antenna (such as the rms servo error in wind, closed loop bandwidth, etc.) through the values of Q and V . A closed-form relationship does not exist between those matrices and the required behavior of an antenna. Thus, an immediate solution is a "trial-and-error" approach. However, there are too many parameters in Q and V to make this approach effective. Moreover, Q and V depend on the choice of the state-space coordinates, with some coordinates more useful than others. Physical coordinates, such as structural displacements, motor torques or currents, although easy to interpret, are highly coupled, therefore create undesirable difficulties. It was verified that the modal coordinates simplify the design because they are weakly coupled (i.e., modifications of one of them weakly impact the remaining ones). In consequence, Q and V matrices are diagonal (therefore there are less parameters to control the closed-loop dynamics). Additionally, we assume the equality of Q and V , i.e., $Q=V$. This simplification eases the search for the "best" controller and is rather opportunistic, with a goal to simplify the GUI approach. Experience shows, however, that the results obtained using this assumption are satisfactory or even exceed the expectations. The reader shall note that in individual cases, the tuning of Q and V separately may improve the performance.

Taking into account the above considerations, the matrices Q and V are diagonal, i.e., $Q=V=diag(q)$, where q is a weighting vector. Note that components of q correspond to the components of the state vector x . Thus, for the state vector as in section 4.1, the weighting vector has the form similar to x

$$q = \{q_{ei} \quad q_e \quad q_a \quad q_{f1} \quad q_{f1} \quad \cdots \quad q_{fn} \quad q_{fn}\}^T$$

where q_{ei} is the weight of the integral of the servo error, q_e is the weight of the servo error, q_a is the weight of the variable x_a , q_{f1} is the weight of the first natural mode, and q_{fn} is the weight of the n th natural mode.

Now, we have to specify $2n+3$ weights, which, for a typical number $n=4$ it makes 11 weights. The critical fact is that the weights have weak dependence amongst each other: the i th modal weight, q_{fi} , impacts mostly the i th mode, i.e., state variables, x_{f1} , and x_{f2} . Also, the “tracking” weights, q_e , q_{ei} , and q_a impact mostly the “tracking” states, e , e_i , and x_a , see Ref.[1]. The design process is illustrated in the next subsection.

4.3. GUI for the LQG Controller Design

The GUI display is shown in Fig.9. The GUI allows for simple manipulations of the design parameters and observations of the antenna performance to make design decisions. Alternative methods involve Matlab code manipulation.

1. On the left-hand side of the interface, there is a short description of the tool. The description falls under the heading ‘LQG Controller for BWG Antenna’. LQG stands for Linear Quadratic Gaussian. This is a model-based controller, meaning the antenna’s *linear* model is a part of the controller. *Quadratic* refers to the index that the controller minimizes, and *Gaussian* refers to disturbances and noises acting on the antenna. BWG is simply a Beam Wave-Guide antenna.
2. The frame below the description contains a “.mat” file (Matlab’s data file). In our case this file holds the A , B , and C matrices described in the Appendix, see equations (A1a) and (A1b). These are the parameters of the antenna rate-loop model. ‘ a ’ is a square matrix, n -by- n . ‘ b ’ is a column vector: n -by-1, and ‘ c ’ is a row vector: 1-by- n . The user must enter **load filename.mat** in the editable text box.
3. We now focus on the eight sliders and their three displays. These are the user tools to modify the controller’s performance. To understand how these play a role in controller design, look back to Section 4.2.

Each slider (except the wind-speed slider) ranges from 0 to 100 (these are dimensionless numbers). By clicking on the end arrows, the marker will move *one tenth* of a slider unit in that direction. Clicking in the slider’s path will move it *one* slider unit in that direction. Alternatively, the user may drag the marker any distance.

To be sure of what value the controller is using, note the numerical value directly to the right of the slider, and the appropriate of the three adjacent displays (for relative weight comparisons). The method of determining where to position the marker along each slider is documented later, in Section 4.4. Also, see the tooltip strings ('Tooltip strings' are the little yellow messages that pop up when the mouse lingers too long over one spot) for a description of which slider effects what most heavily. If you choose to go on without understanding the cause and effect relationships of this controller, the best advice we can offer is to start with all the weights at zero and to *slowly* adjust them.

4. In the bottom right corner of the GUI screen are the simulation results and the values of the proportional and integral gains. Each of the result headings will display a one-sentence description of what it represents in a 'tooltip string'. Each of the values below are updated after the 'Simulate' button is pressed.
5. The upper-left plot is the step response to a one-degree offset. The upper-right plot is the magnitude of the transfer function of the closed-loop system. The axes are labeled, and between the two plots there is a legend.
6. The "simulate" button is used to execute the simulation with the parameters on the screen.

The antenna performance can be observed on the GUI display, see Fig.10a,b:








- *Settling time* refers to the time it takes for the step response to get (and stay) within 3% of the commanded response. The two gray lines in the upper left-hand plot represent this $\pm 3\%$ range. In the event that the user creates an unstable controller, this value will reference her/him to the Matlab command window (which must be open anyway), where a message will be printed to the screen describing the model's instability.
- *Bandwidth* is the frequency at which the magnitude of the transfer function moves below -3dB (or below 0.71). The 0.71 value is marked in gray on the upper right plot.
- Not surprisingly, *overshoot* is the percent that the antenna overshoots its commanded value.
- *rms (root mean square) error* indicates the servo error value in wind gusts for the selected wind speed.
- Steady state *rate offset error* is the difference between the actual antenna position and the commanded position when the antenna is moving with a constant rate. This value shall be zero under steady-state conditions.
- *Maximum disturbance* is the maximum value of the antenna response to the unit step disturbance.

Below these six outputs the proportional and integral *gains* are displayed, which are related, non-linearly, to the proportional and integral *weights*. These values are of

importance because from them, the experienced user may gather information about the controller's robustness.

The performance of the antenna depends on *all* controller gains, or *all* controller weights. The relationship is not simple, and depends on the coordinates the antenna model is represented in. Generally, changing a single weight impacts more than one performance features. If one wants to change a single feature, e.g. expand the bandwidth, one needs to change many weights. However, in the *modal* coordinates of the antenna model each weight influences predominantly one or two performance features, and is weakly coupled with the remaining ones. This helps to follow the improvement of the antenna performance, since each slider predominantly addresses a single performance feature (such as bandwidth, vibrations, overshoot, etc.). The table below summarizes these relationships, while Figure 11 shows the relationships between the step response and the magnitude of the transfer function.

TABLE 1. Relationship between LQG weights and the performance

Slider movement (LQG weight)		Result (performance)
Proportional weight		Settling time, bandwidth, disturbance rejection
Integral weight		Overshoot, rate offset error, disturbance rejection
Special weight		This slider move in emergency only
Frequency weight 1		Vibration amplitude of the lowest (fundamental) frequency, height of the first resonance peak
Frequency weight 2		Vibration amplitude of the second frequency, height of the second resonance peak
Frequency weight 3		Vibration amplitude of the third frequency, height of the third resonance peak
Frequency weight 4		Vibration amplitude of the fourth frequency, height of the fourth resonance peak

4.4. DSS26 Antenna Example

The following steps correspond to the interfaces pictured below in Figs.12a-f. These steps reflect antennas common features, and can be followed as a guideline to the controller design process:

- 0. The design starts with the input of the state space representation of the open-loop antenna model.** For example, the DSS26 antenna representation (A, B, C, D) is a file `abc.mat` located at the directory `c:/matlabr11/lqg/gui/az`. In the lower left window we enter: `load c:/matlabr11/lqg/gui/az/abc.mat`.

1. Next, we select arbitrary, but small, values of proportional and integral weights (say $q_e=0.1$ and $q_i=0.3$) and zero for special and frequency weights see Fig.12a. After simulation the step response shows excessive settling time and overshoot (8.52s and 18% respectively), visible flexible oscillations and an unacceptable maximal disturbance step value (0.82 deg). The magnitude of the transfer function shows low bandwidth (0.21 Hz), sharp resonance peaks, and a larger-than-desired magnitude of the disturbance transfer function.
2. The most excessive oscillations come from the first (or fundamental) mode. Therefore **the weight of the first flexible mode is increased to 5, and other weights are unchanged**, as shown in Fig.12b. The results are visible in the step responses where the oscillations are now invisible, and in the transfer function where the first resonant peak disappeared. Other parameters remained unchanged except for the disturbance responses that deteriorated, and the second resonance peak is excessive.
3. **The proportional weight is increased to 10**, c.f., Fig.12c. This move reduced the settling time to 3.8s, overshoot to 3.6%, and expanded bandwidth to 0.6 Hz. The maximum disturbance step response is lowered to 0.49deg, but the second resonance peak has raised to a dangerous level.
4. **The integral gain was increased to 3**, refer to Fig.12d. While bandwidth was expanded to 0.8 Hz and maximum of the disturbance step response is reduced to 0.41, overshoot and settling time increased to 13% and 4.0s respectively.
5. In this step the second resonance peak is damped, by **increasing the “Frequency Weight 3” to 10** see Fig.12e. The step responses are smoother, the second resonance peak in the magnitude of the transfer functions is reduced.
6. **In the final step, the proportional weight is increased to 25 and integral weight is increased to 12**, c.f., Fig.12f. The settling time decreased (3.0 s), but overshoot increased (14%). The maximum step disturbance response decreased to 0.38 deg and the bandwidth significantly increased to 2.2 Hz. This is an acceptable solution and in order to save the weights governing the output, we hit the ‘Weights’ button and the values are printed to the Matlab command window. Now we close the design window and begin the next stage of design, the fine tuning of the controller.

5. Fine Tuning of the LQG Controller

Certain features of the controller obtained through the approach described above can be improved by further modifying the controller weights. For example, the oscillations of the closed-loop response can be further reduced or the closed-loop response can be modified such that the wind disturbance does not exceed its specification. However, at this stage, the closed-loop states are not as easily decoupled as the open-loop states, therefore the relationship between the weights and the closed-loop dynamics is less clear,

and often difficult to track. For this reason a constrained optimization approach is used to tune the already designed controller.

5.1. Description of Variables

In this approach the weights are design parameters, and the following variables are either optimized or constrained:

1. steady state servo error in the 0.1 deg/s rate offset, e_r (mdeg), defined as $e_r = \|r(t) - y(t)\|_2$, for $18 < t < 20$ s.
2. max value of the servo error due to the unit step disturbance (deg)

$$e_{dmax} = \max(|e_d|), \text{ for } t \geq 0$$

3. rms servo error in 15 mph wind gusts, e_w (mdeg), for $0 \leq t \leq 100$ s
4. overshoot of the unit step response, e_o (%)
5. settling time of the unit step response, t_s (s)
6. bandwidth, f_o (Hz)
7. magnitude of the closed-loop transfer function (from the command to the encoder) for high frequency range (above 3.5 Hz), that is $m_h = \max m(f)$, for $f \geq 3.5$ Hz
8. magnitude of the closed-loop transfer function (from the disturbance to the encoder), i.e., $m_{dmax} = \max m_d$, for $f \geq 0$ Hz.

The index (a positive function to be minimized) is defined as follows:

$$f = w_1 e_r + w_2 e_{dmax} + w_3 e_w + w_4 e_o + w_5 t_s + w_6 (3 - f_o) + w_7 m_h + w_8 m_{dmax}$$

Each variable in the function to be minimized is weighted (or multiplied by a positive number). The weight w_i , indicates the relative importance of each variable. If $w_i = 0$ the i th variable is not optimized. In the above-defined function, the bandwidth is maximized by using $3 - f_o$ variable rather than the bandwidth f_o itself. For BWG antennas the bandwidth will not exceed 3 Hz, therefore the minimization of $3 - f_o$ leads to expansion of f_o .

The variables above are functions of the LQG weights, q_i , $i=1, \dots, n$, therefore f is function of q as well. The initial values of the LQG weights are taken from the design part of GUI, as described in Section 4.

5.2. GUI Description

This tool is quite similar to the previously described tool, see Fig.13. The two plots of the fine-tuning GUI are identical to those of the design GUI except that the step plot

has a ten second span of x-axis in the fine-tuning GUI. The user is given a line on which she/he enters the file containing the discrete time state space representation of the rate-loop model (either azimuth or elevation axis).

In this GUI, there are eight variables explained above that can be either minimized, or constrained. Different variables and their changes are displayed and since the start can be tracked throughout. The user may also view the initial variable values, the iteration number, and the function value in the Matlab command window, see Fig.14.

The fine-tuning GUI is meant only as a small improvement tool. This restriction follows from the fact that the program searches for local minimum only, hence dramatic improvement is not expected, unless the initial design is a lousy one. The requirement to dramatically improve the design (by setting demanding constraints for example) usually leads to the termination of the program with the message “no feasible solution exists” in the Matlab command window.

The Matlab optimization function, *constr*, is used here to find a local minimum of a constrained, nonlinear, multivariable function. Before attempting to minimize a function, it will ensure that the constraints have been met. Two main decisions must be made when using the *constr* function. One is whether to optimize or constrain a certain variable. The other is how many iterations are necessary to run in a block. Too many iterations may require long simulation time and give the user the feeling of being out of control, too few requires frequent re-starting.

5.3. Fine-Tuning Procedure

The following procedure is recommended to tune the controller using the GUI:

1. For each variable **verify the starting values** by clicking on the ‘optimization weights’ (rather than ‘constrain’) radiobutton, and set the weights to zero. Construct the weight vector:

$$\begin{bmatrix} \textit{Integral} \\ \textit{Proportional} \\ \textit{Special} \\ \textit{Frequency1} \\ \textit{Frequency2} \\ \textit{Frequency3} \\ \textit{Frequency4} \end{bmatrix}$$

using the last values from the design GUI. (Using arbitrary initial values will most likely result in an ‘infeasible’ result.) Run the optimizer by setting maximum number of iterations to 1 and by pressing the ‘Optimize’ pushbutton.

2. Keep all of the ‘optimize’ radiobuttons on, but **scale the different variables according to your priorities**, i.e. change the weights to nonzero prioritized values.

Run this optimization in increments of about 300 iterations, or until the system is optimized.

Note: If the system does not find the optimal solution on the first run through, be sure to keep the weight vector (pushbutton above the weight vector display) to begin at the previous finish point. To use these values the 'Keep' pushbutton must be pushed before the 'Optimization' pushbutton.

3. After the system is optimized ('Converged Successfully' is printed in the Matlab command window), look through the values of the variables and **decide which performance values are acceptable** and which are not. Of the ones that are not, select the least acceptable. Starting with the least acceptable variable, choose its 'constrain' radiobutton and select a constraint value that is tighter than the current value, but loose enough so that the solution is feasible. If infeasible values are selected, the algorithm will let the user know right away. Refresh the weights if necessary. Again, reset the weight vector to pick up where it left off and run the optimization until it converges successfully.
4. Repeat the previous step (not forgetting to reset the weights vector before pushing the 'Optimize' button), **constraining each unacceptable variable until the desired output is achieved.**

Because it is the gains (K_i , K_p , K_f , and K_e) that are needed to implement a controls design, these values are saved (and replace previous values) after each block of iterations as `gains.mat` in the working directory.

5.4. DSS26 Antenna Example

We start with the final design of Section 4 and follow the procedure in Section 5.3. The steps below correspond to the interfaces pictured in Figs.15a-d. The goal is to improve the design by increasing damping of the higher frequency oscillations (the vibrations are visible in the step response) and to improve the rms error due to wind disturbances. Other beneficial results were obtained as a byproduct of our efforts.

1. **Verify the starting values** with the weight vector equaling:

$$\begin{bmatrix} 12 \\ 25 \\ 0 \\ 5 \\ 0 \\ 10 \\ 0 \end{bmatrix}$$

The result is shown in Fig.15a, which is identical to Fig.12f in section 4.

2. **Prioritize your goals.** We wanted to minimize the rms error, so we assign the largest weight of 1.0 to this value. Then, we wanted to improve the damping for the high frequencies (and additionally the disturbance transfer function, the settling time and maximum value of disturbance), therefore we selected 0.1 weights for these variables. We were satisfied with bandwidth, and rate error, therefore we weighted these with 0.01. Overshoot was not of concern for our purposes, so we assign it a weight of 0.001. We chose a maximum of 300 iterations. At the end of these 300 iterations, the system had still not converged, so we started with the weight matrix from the previous iteration and ran the system again for the next 300 iterations. This time it converged after about 100 iterations. The optimization results are visible in Fig.15b, which shows improved values of all variables, except overshoot.
3. **Review the performance.** We reviewed the performance and decided that the maximum disturbance was unacceptable therefore we constrained it to 0.34 at the same time relaxing weighting of overshoot, settling time and bandwidth values. Again we set the maximum number of iterations to 300 and the system converged after about 200 evaluations. The results are shown in Fig.15c, where the constraint was met and rms error, bandwidth, and overshoot were improved. Command transfer function at high frequencies, disturbance transfer function, and settling time deteriorated.
4. **Final corrections.** As a last step we constrained overshoot to 25% leaving everything else as it was in step (3). The result after completing 300 iterations and another 100 iterations is visible in Fig.15d, with an overshoot of 23.3% and additionally improved disturbance transfer function, settling time and maximum disturbance.

After completing these four steps, we were satisfied with the results except for bandwidth, which would have preferably been over 2.2 Hz. The next step involved constraining bandwidth to this value and leaving all other settings as they were. The result was an infeasible request. We next tried decreasing the optimization weights, but came up again with an infeasible outcome. At this point we decided that the results actually were quite 'fine-tuned' and called it quits.

6. Conclusions

This paper describes two types of GUI that help design LQG controllers for the antennas and radio-telescopes: the LQG controller design GUI, and a GUI for the fine-tuning of the LQG controller design. In the *design* GUI the user simply manipulates controller parameters to observe their impact on the closed-loop system performance (such as overshoot, settling time, bandwidth, rate-offset error, maximum response to step disturbance and response to wind gusts). A procedure is given that guides the user towards achieving a particular goal. The *fine-tuning* GUI improves the already designed

controller by running an optimization procedure. The closed loop performance besides the ones mentioned above include vibration damping and maximum magnitude of the disturbance transfer function, and are either optimized or constrained, so that a desired performance is approached. A procedure is given to guide the user towards achieving performance requirements.

7. Acknowledgements

The research described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

References

1. W. Gawronski, and J.A. Mellstrom, "Control and Dynamics of the Deep Space Network Antennas," in: *Control and Dynamics Systems*, ed. C.T. Leondes, vol. 63, Academic Press, San Diego, CA, 1994, pp. 289-412.
2. W. Gawronski, C. Racho, and J.A. Mellstrom, "Application of the LQG and Feedforward Controllers for the DSN Antennas," *IEEE Trans. on Control Systems Technology*, vol.3, 1995.
3. W. Gawronski, W. H. G. Ahlstrom, Jr., and A. B. Bernardo, "Design and Performance of the DSS-14 Antenna Controller," *TMO Progress Report*, vol. 42-135, 1998.
4. W. Gawronski, *Dynamics and Control of Structures: A Modal Approach*, Springer-Verlag, New York, 1998.
5. F.W. Fairman, *Linear Control Theory*, Wiley, Chichester, 1998.
6. S. Skogestad, I. Postlethwaite, *Multivariable Feedback Control*, Wiley, 1996

APPENDIX. Derivation of the LQG Gains

In the LQG design process the controller gain (K_c), and the estimator gain (K_e) are determined. The LQG controller block diagram is shown in Fig.A1. It consists of the antenna state space model (A, B, C), from which the antenna rate-loop equations are generated

$$\dot{x} = Ax + Bu + v. \quad (\text{A1a})$$

$$y = Cx + w \quad (\text{A1b})$$

In this model the antenna state vector is denoted x . The antenna is perturbed by random disturbances: the input noise, v , and the output (or measurement) noise w . The input noise is predominantly wind disturbance, and has covariance V . The measurement noise has covariance I . The assumption of unit covariance of the measurement noise does not impact the final design results, since proper scaling of the input disturbance can compensate non-unit covariance value. It is assumed that the input and output noises are not correlated. This assumption is equivalent to independence of their sources. Indeed, the measurement noise is independent of the wind disturbances.

The estimator evaluates antenna states, using the rate input (u) and the encoder output (y) of the antenna. The estimated state vector is denoted \hat{x} , and the error between the actual encoder output and the estimated output is defined as $\varepsilon = y - C\hat{x} = y - \hat{y}$. The estimated state is obtained from the following equation:

$$\dot{\hat{x}} = A\hat{x} + Bu + K_e \varepsilon. \quad (\text{A2})$$

One can see that the estimated state is proportional to the estimation error; the proportionality gain K_e is called the estimator gain.

The controller forms the negative feedback between the estimated state \hat{x} and the antenna input u

$$u = -K_c \hat{x}, \quad (\text{A3})$$

where the controller gain matrix K_c .

The task is to determine both the controller and estimator gains such that the performance index J is minimal, where

$$J^2 = E \left(\int_0^\infty (x^T Q x + u^T u) dt \right) \quad (\text{A4})$$

and Q is given, positive semi-definite state weight matrix.

It is known, see Refs.[5,6], that the minimum of J is obtained for the controller gain matrix obtained as

$$K_c = B^T S_c, \quad (\text{A5})$$

where B is the input matrix of the antenna model, and S_c is a solution of the following Riccati equation

$$A^T S_c + S_c A - S_c B B^T S_c + Q = 0. \quad (\text{A6})$$

Similarly, the optimal gain in the estimator equation (A2) is given as follows

$$K_e = S_e C^T, \quad (\text{A7})$$

where C is the output matrix of the antenna model, and S_e is the solution of the following algebraic Riccati equation

$$A S_e + S_e A^T - S_e C^T C S_e + V = 0. \quad (\text{A8})$$

Finally, the controller gain is split unto proportional, integral, and flexible mode gains

$$K_c = [k_i \quad k_p \quad K_f] \quad (\text{A9})$$

to fit into the configuration as in Fig.8.

FIGURE CAPTIONS:

Figure 1. NASA/JPL beam wave-guide antenna

Figure 2. Open-loop (or rate-loop) antenna model

Figure 3. Simplified rate-loop model

Figure 4. Step response overshoot and settling time (upper figure), and transfer function bandwidth (lower figure).

Figure 5. Bode plots of the DSS26 azimuth rate-loop model

Figure 6. Rate offset command (dashed line), and the antenna encoder reading (solid line)

Figure 7. Antenna servo error in 20-mph wind gusts.

Figure 8. Structure of the LQG control system

Figure 9. Interface of the LQG design

Figure 10. Design GUI displays: a) step responses, b) magnitudes of the transfer functions

Figure 11. Properties of the step response and the magnitude of the transfer function

Figure 12a. Interface 1, Example 4.4.

Figure 12b. Interface 2, Example 4.4.

Figure 12c. Interface 3, Example 4.4.

Figure 12d. Interface 4, Example 4.4.

Figure 12e. Interface 5, Example 4.4.

Figure 12f. Interface 6, Example 4.4.

Figure 13. The fine-tuning interface

Figure 14. The fine tuning Matlab command window:

- (a) Weight vector of the design parameters
- (b) Number of function evaluations before breaking
- (c) See section 4.1 below
- (d) Initial values of the variables
- (e) 'constr' output:
 - f-count -- Iteration number
 - function -- Value of function to be minimized
 - max{g} -- Maximum constraint value
 - step -- Step size
 - procedures -- Infeasible...Hessian modified...etc.
- (f) Infeasible, Successful convergence, Maximum number of iterations

Figure 15a. Interface 1, Example 5.4.

Figure 15b. Interface 2, Example 5.4.

Figure 15c. Interface 3, Example 5.4.

Figure 15d. Interface 4, Example 5.4.

Figure A1. The block diagram of the LQG controller: A – antenna (hardware); E – estimator (software); PI – PI controller (software); F – flexible mode controller (software). Triangle denotes integration

Variables:

y	–	encoder output
u	–	antenna rate input
r	–	command
e	–	servo error
e_i	–	integral of the servo error
x	–	antenna state
\hat{x}	–	estimated antenna state
ε	–	estimation error
v	–	disturbances
w	–	measurement noise

Blocks:

K_e	–	estimator gain
k_p	–	proportional gain
k_i	–	integral gain
K_f	–	flexible mode gain
A	–	state matrix
B	–	input matrix
C	–	output matrix
C_f	–	flexible mode output matrix

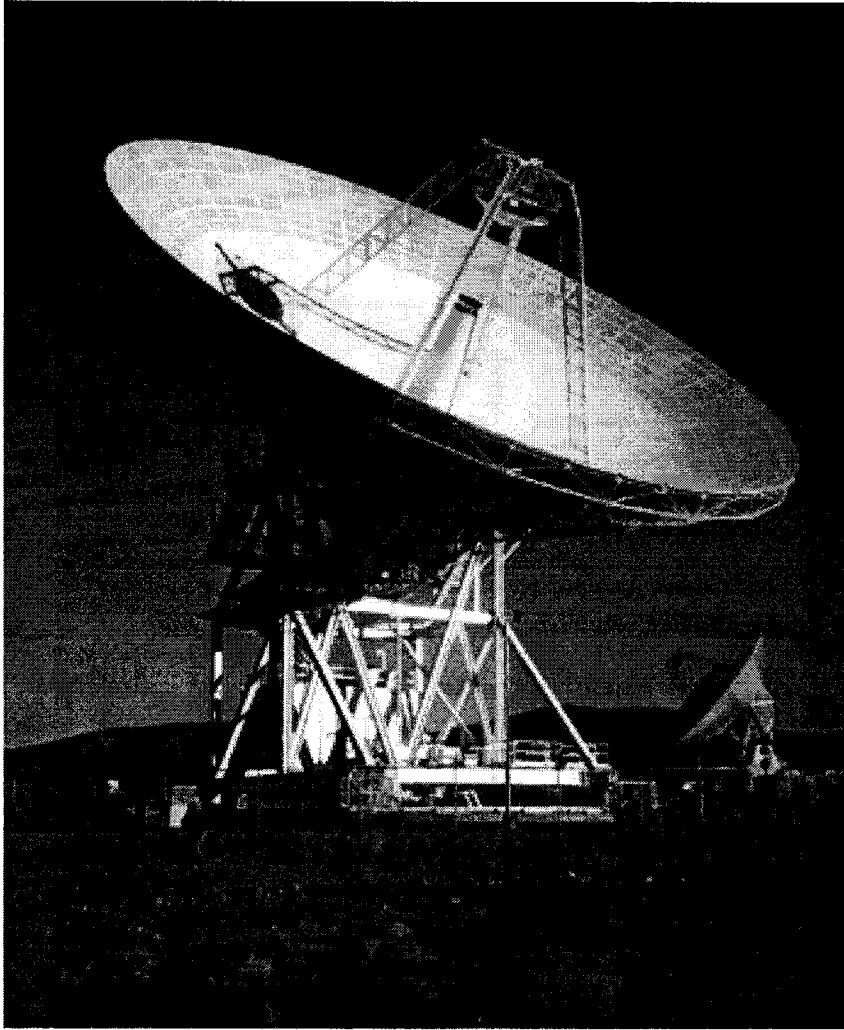


Figure 1.

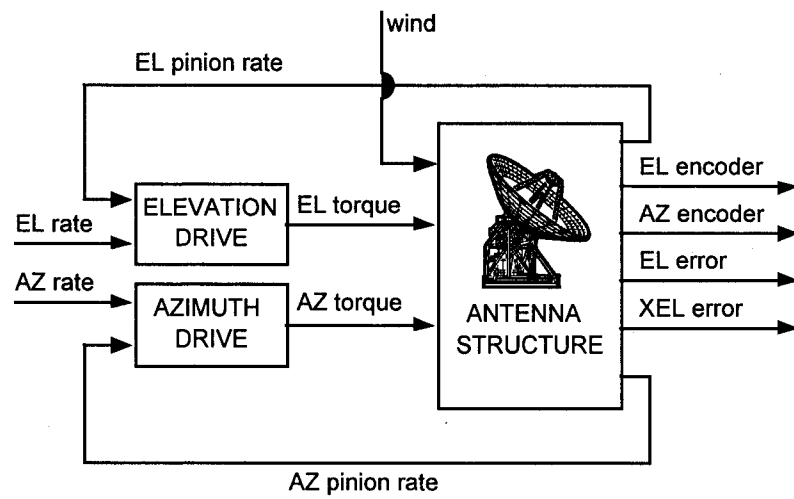


Figure 2.

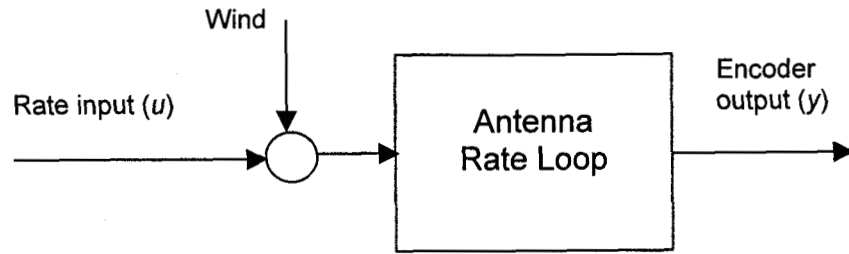


Figure 3.

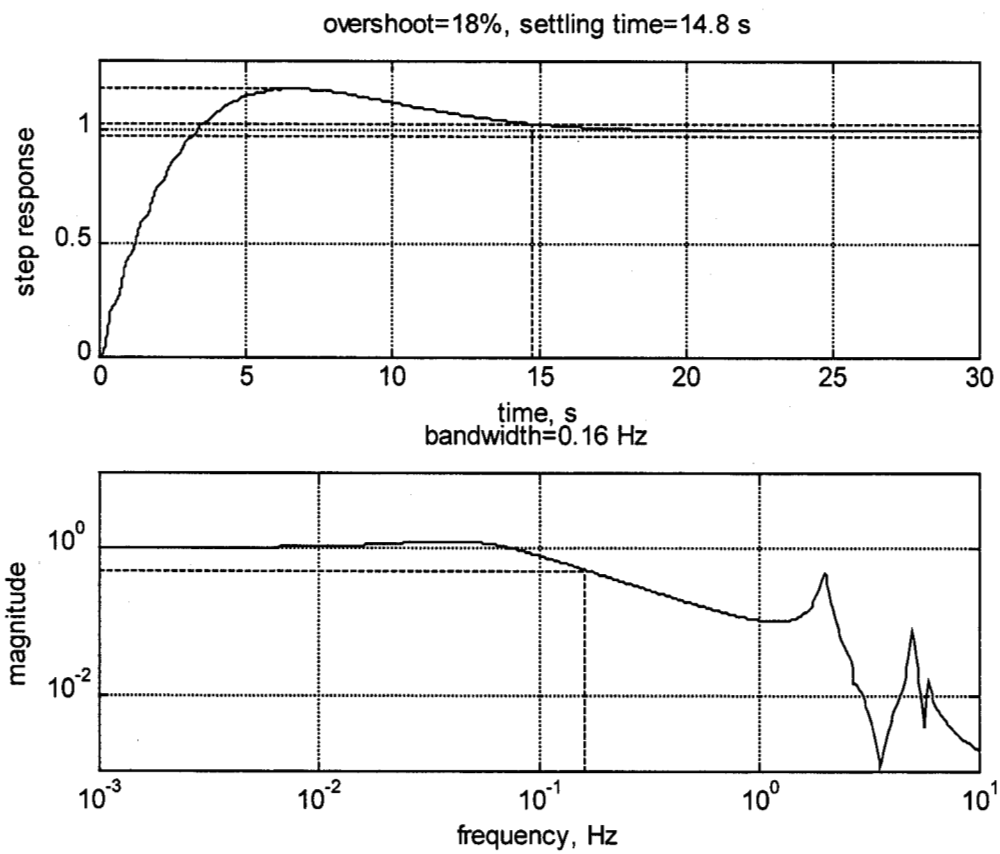


Figure 4.

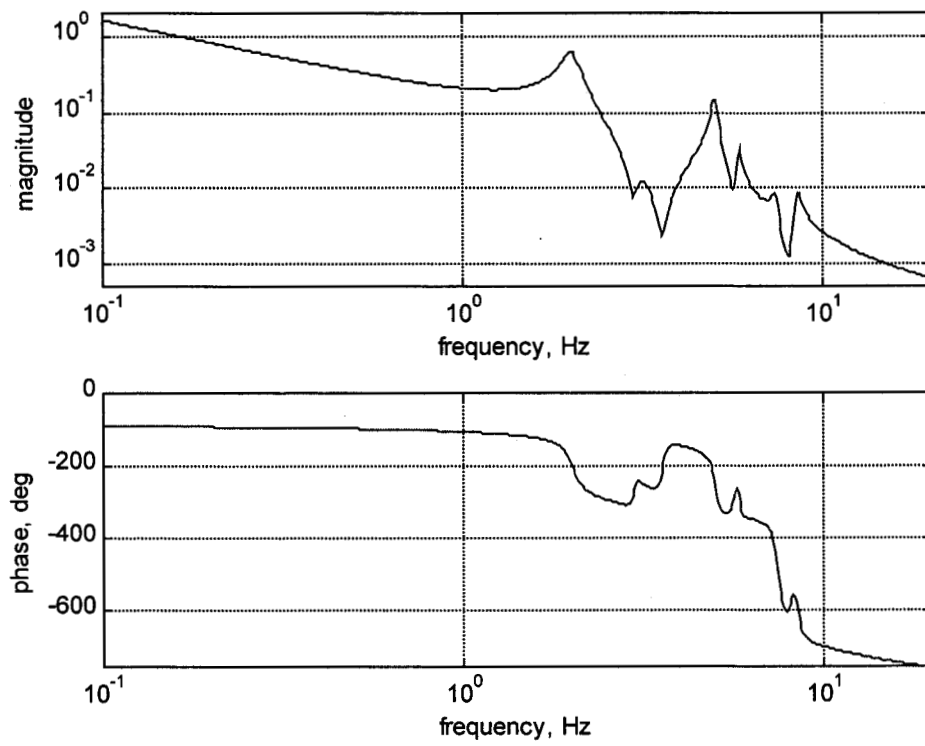


Figure 5.

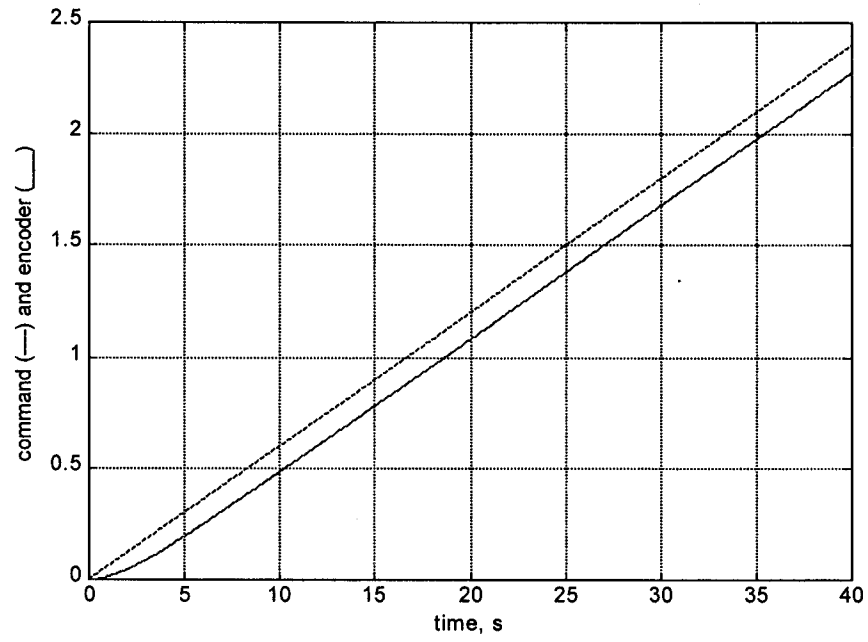


Figure 6.

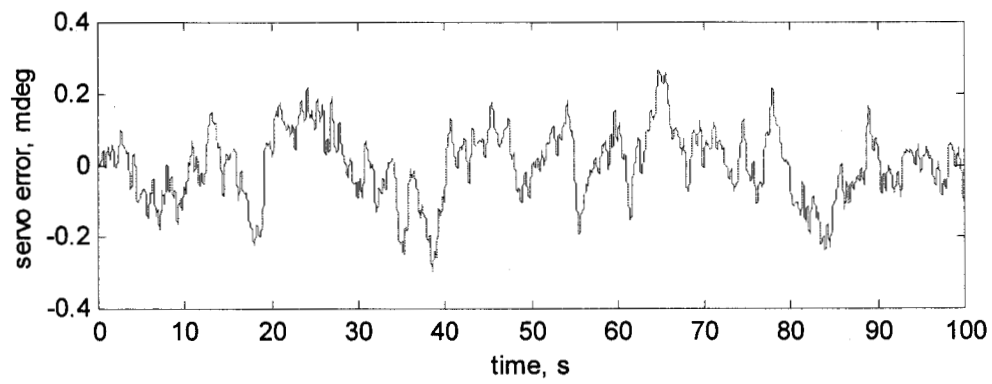


Figure 7.

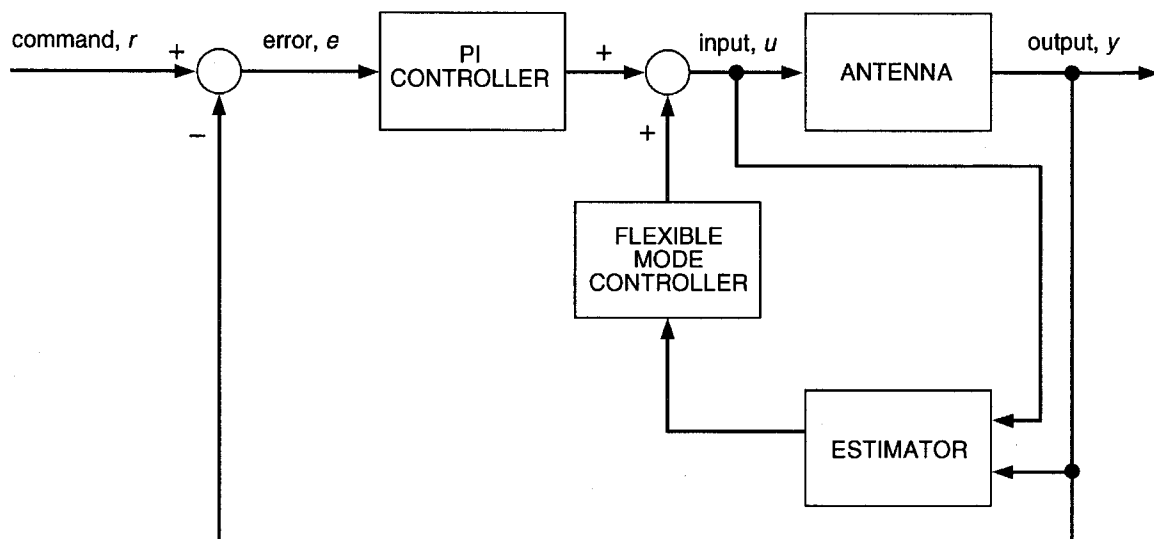
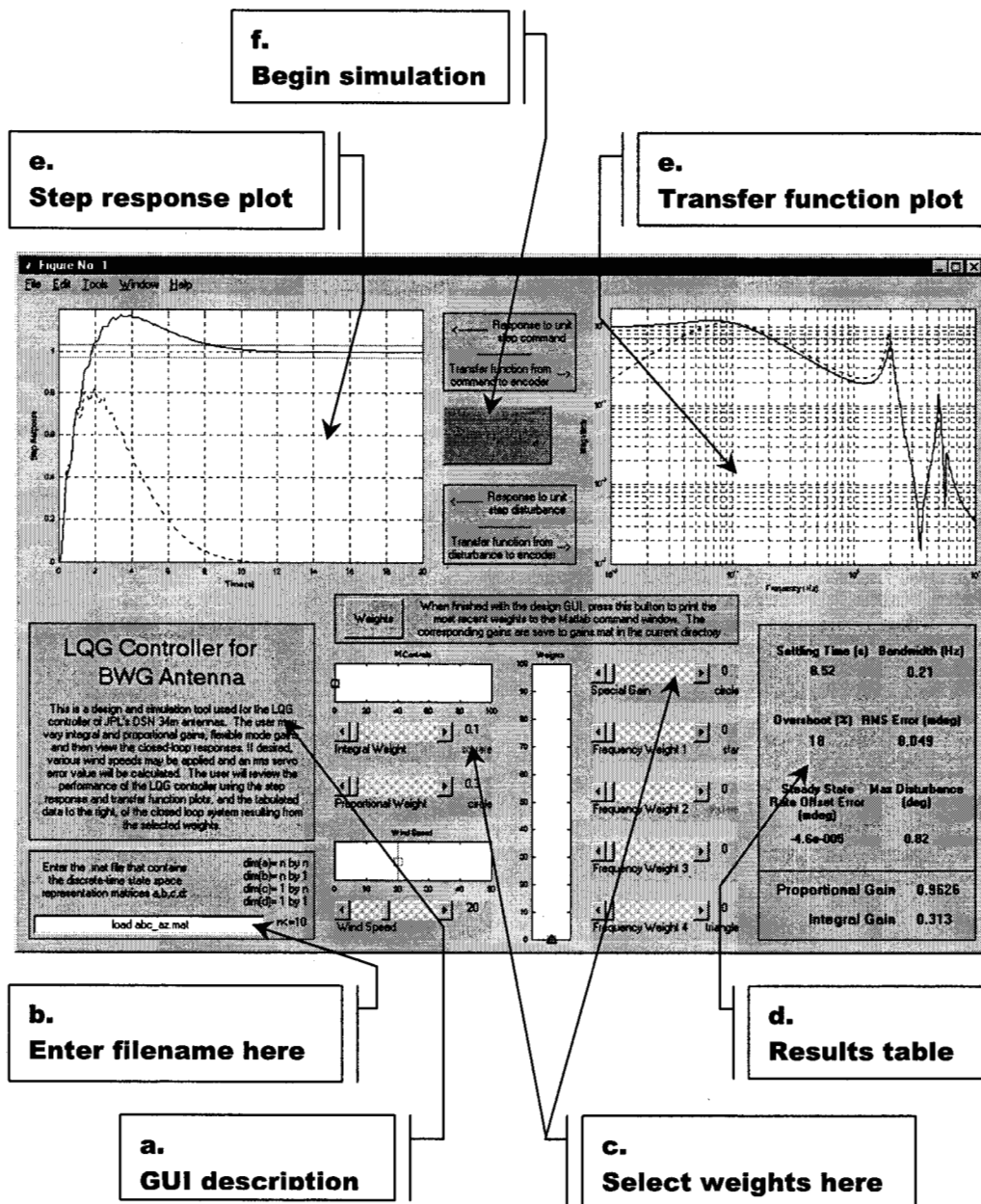


Figure 8.



1. load filename.mat in the editable text box.

Figure 9.

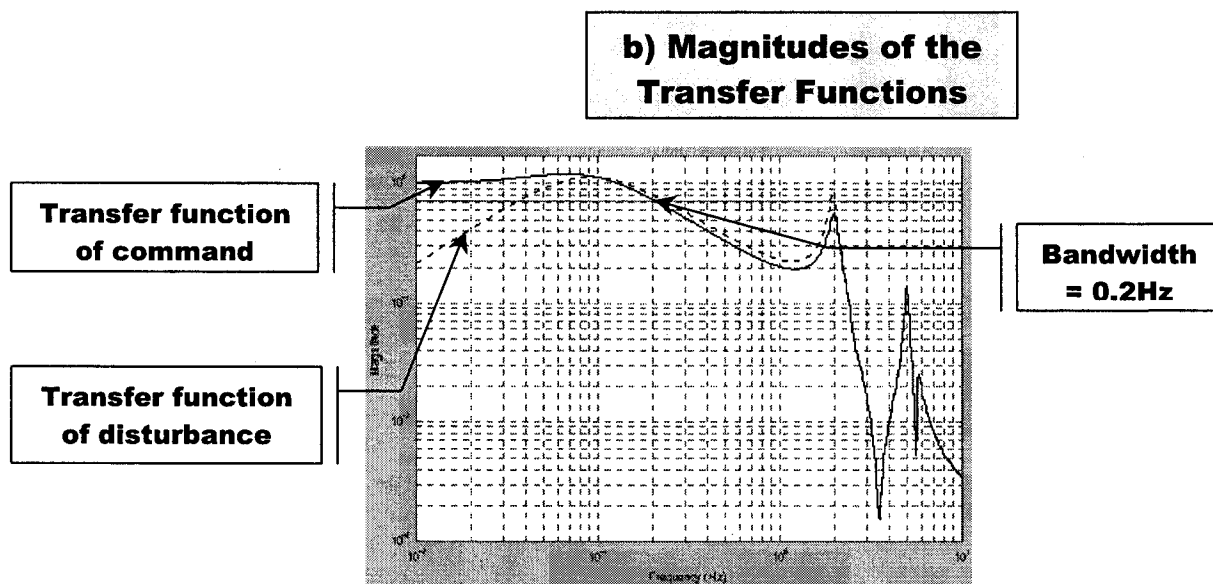
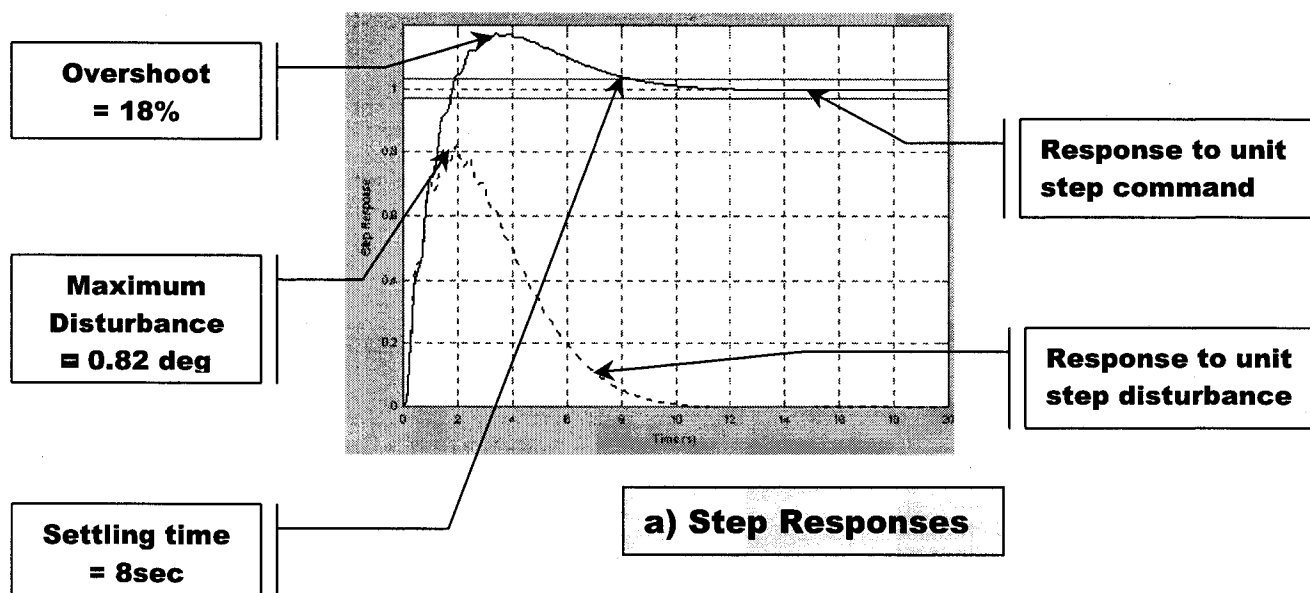


Figure 10.

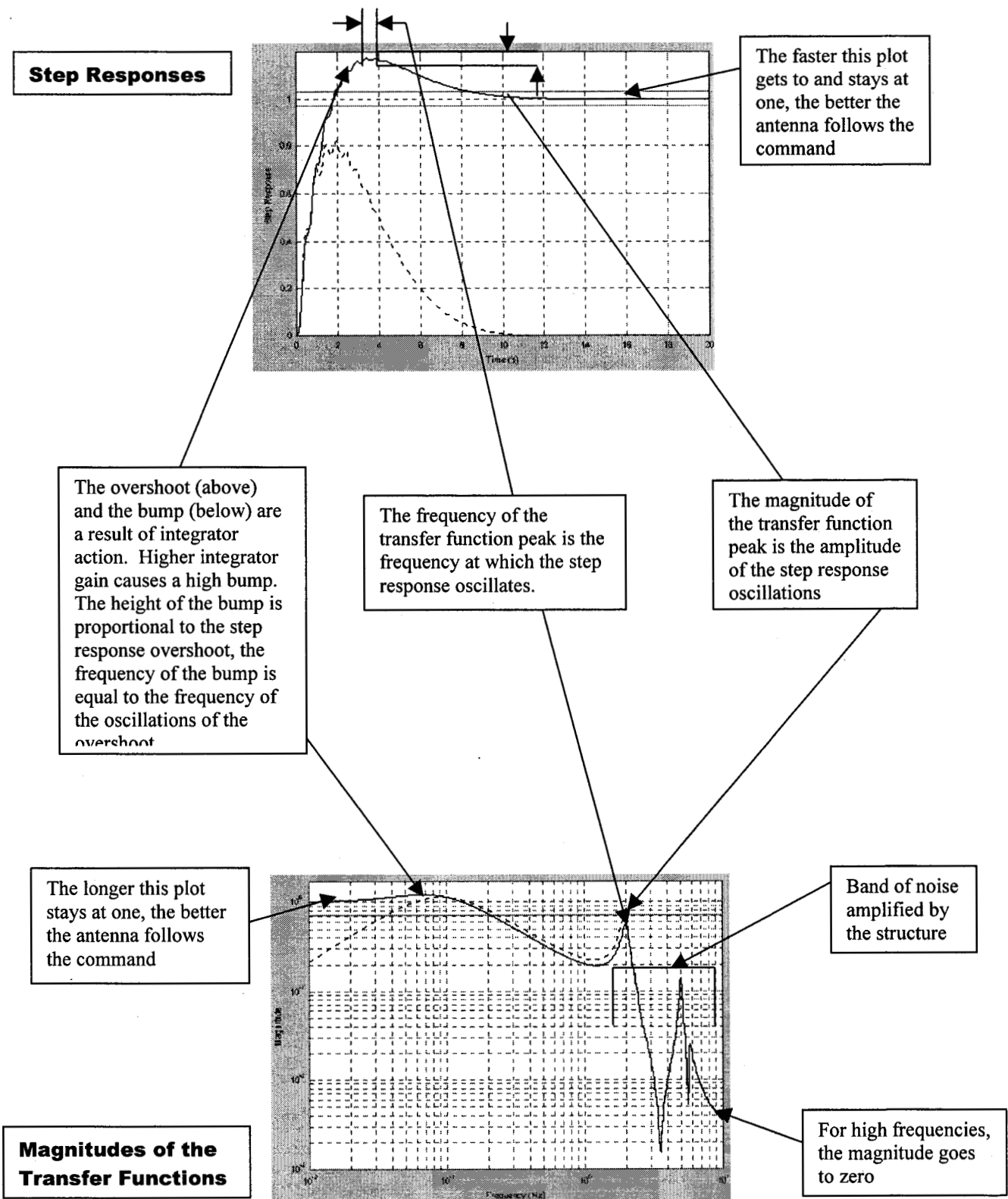


Figure 11.

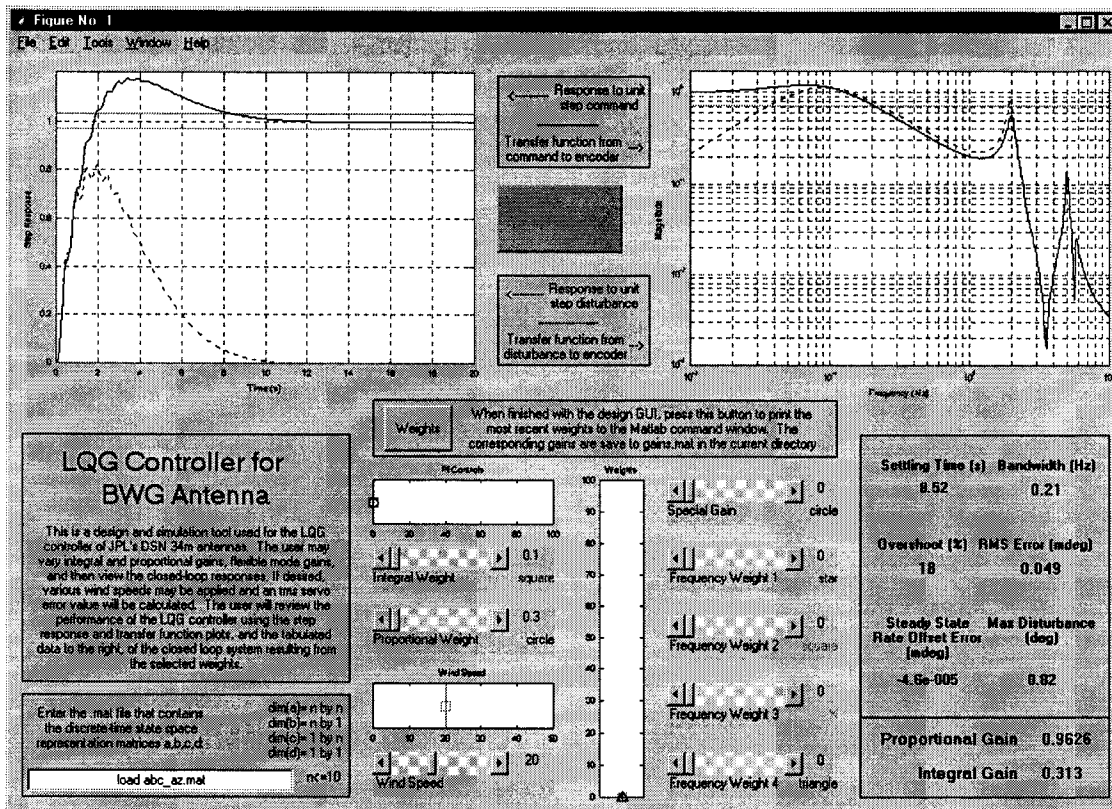


Figure 12a.

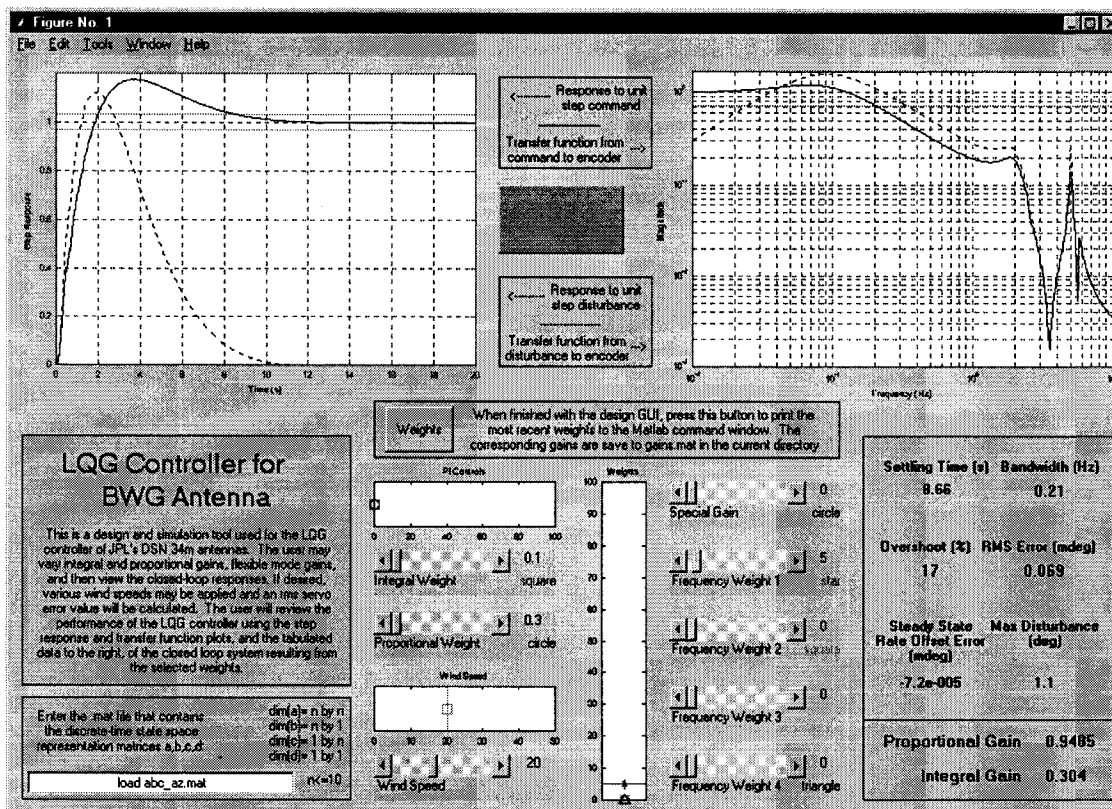


Figure 12b.

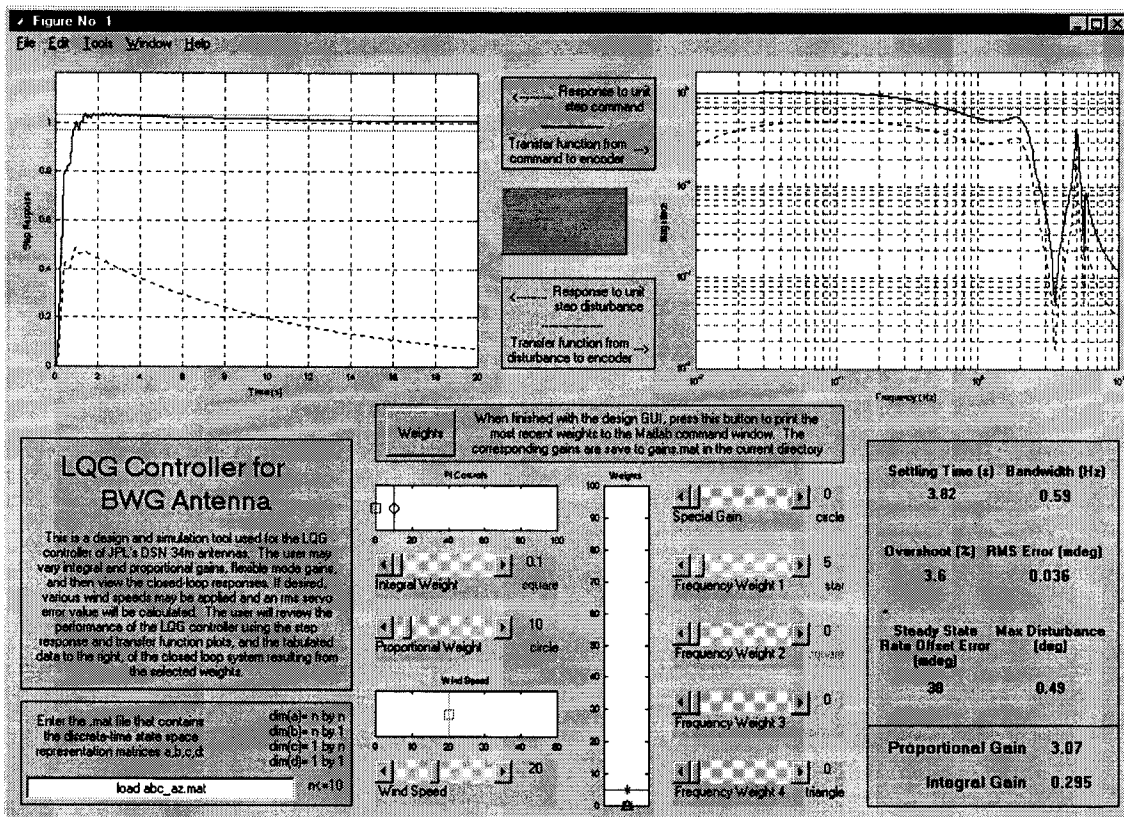


Figure 12c.

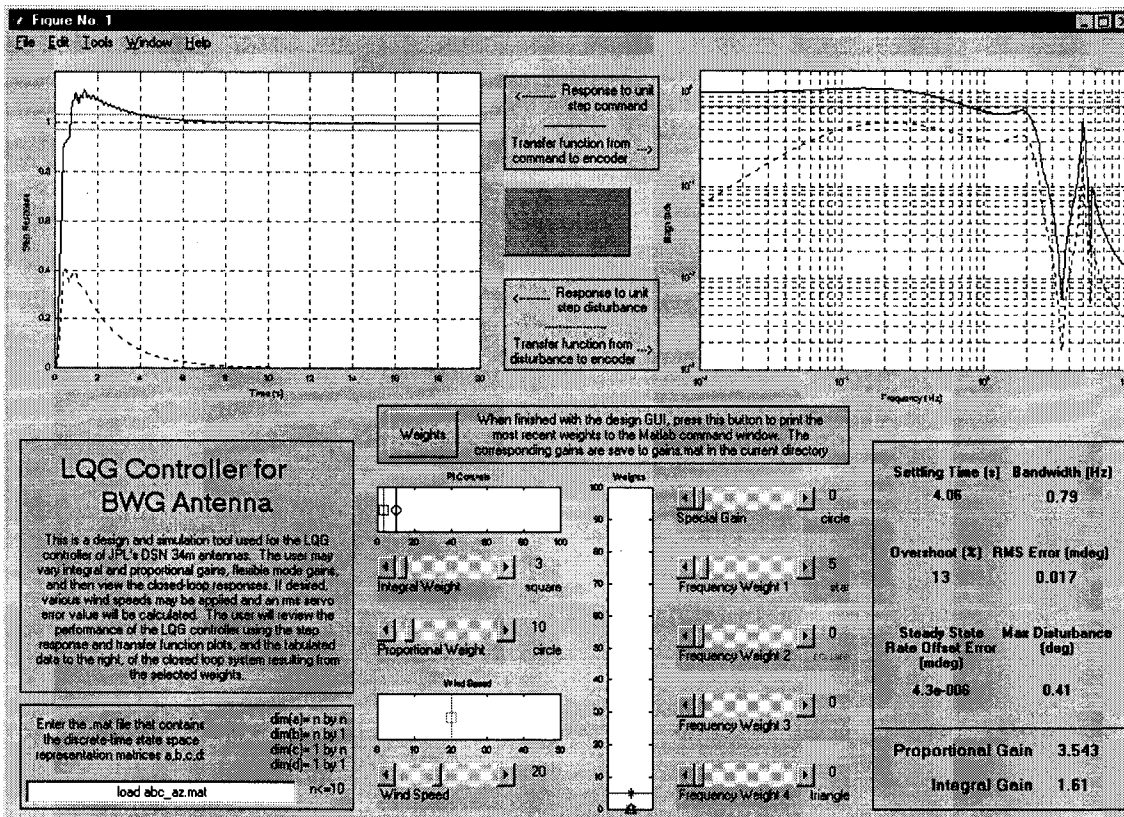


Figure 12d.

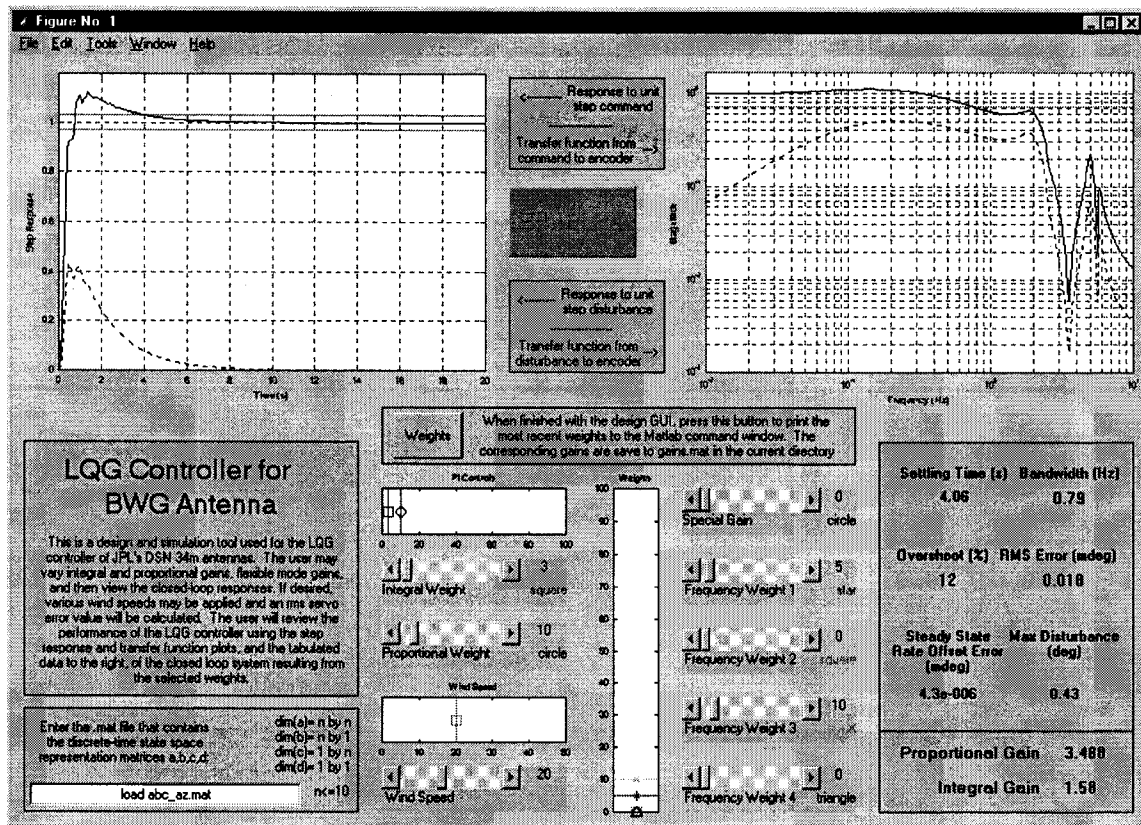


Figure 12e.

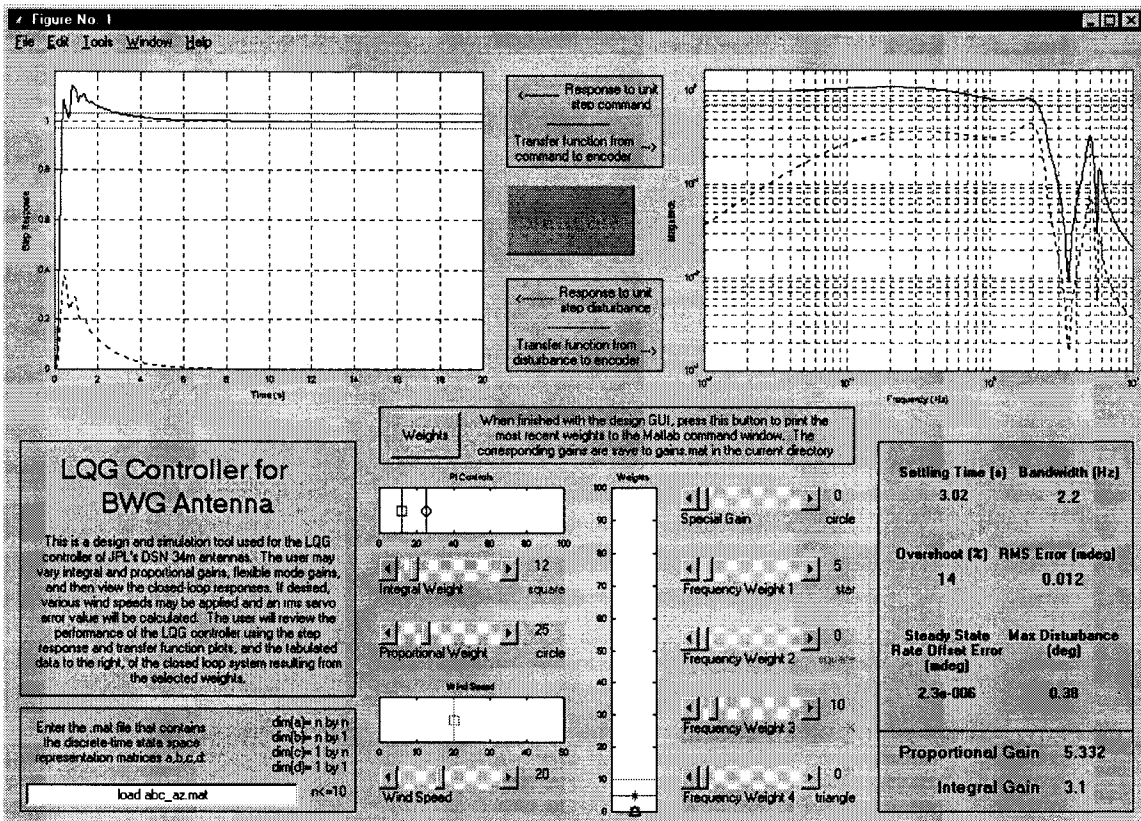


Figure 12f.

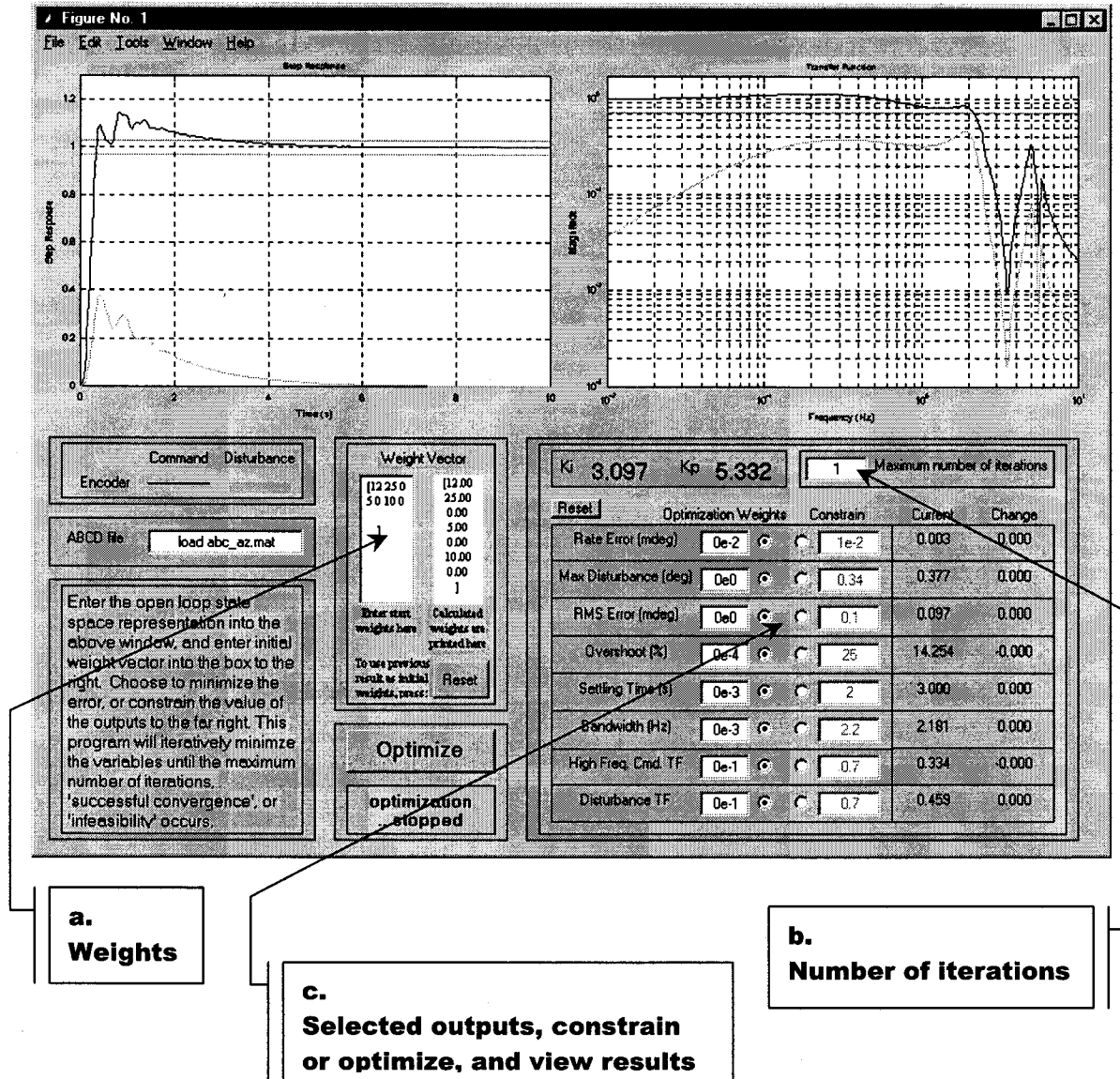


Figure 13.

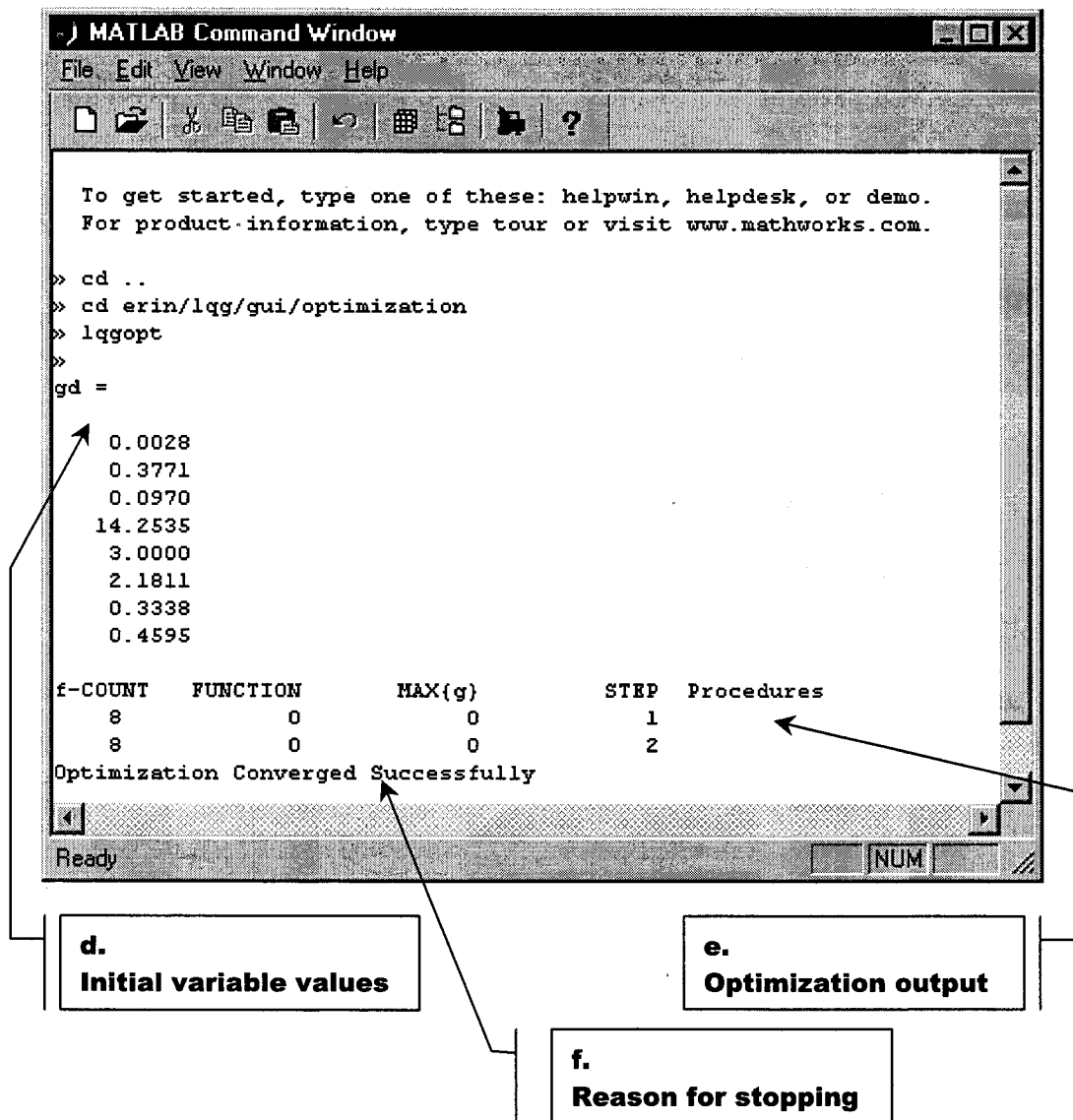


Figure 14.

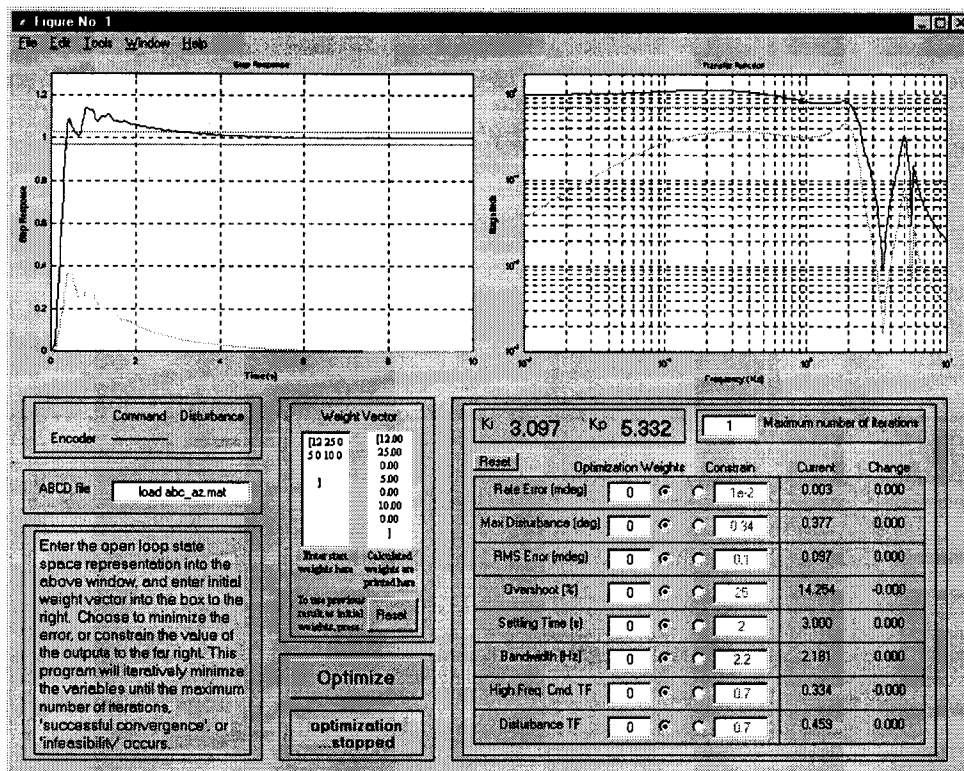


Figure 15a.

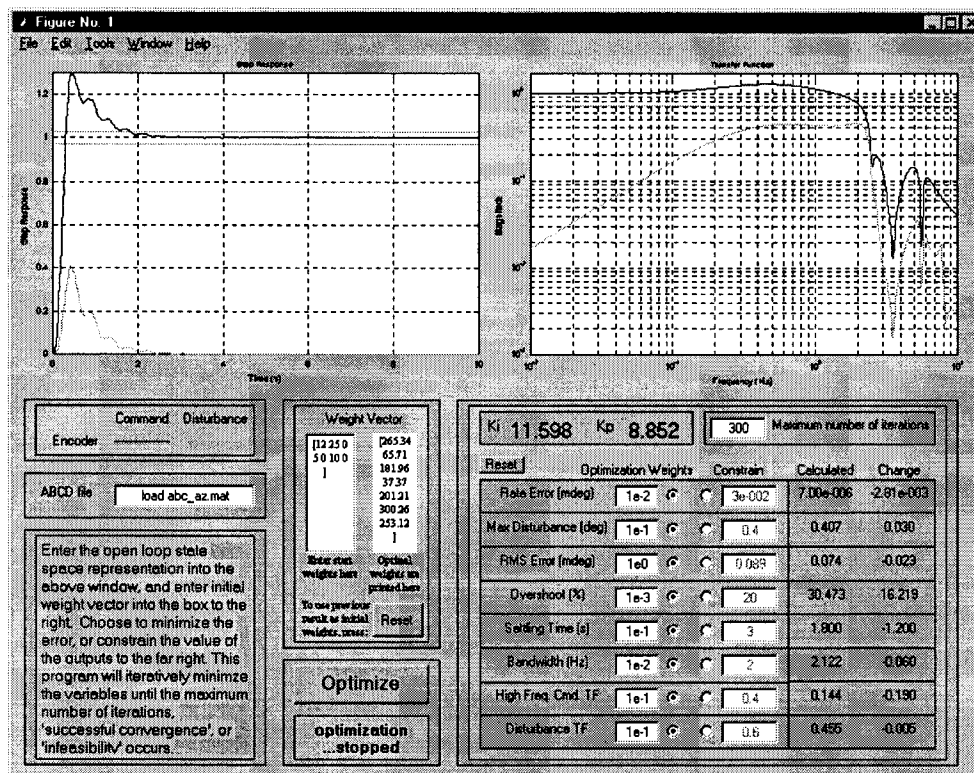


Figure 15b.

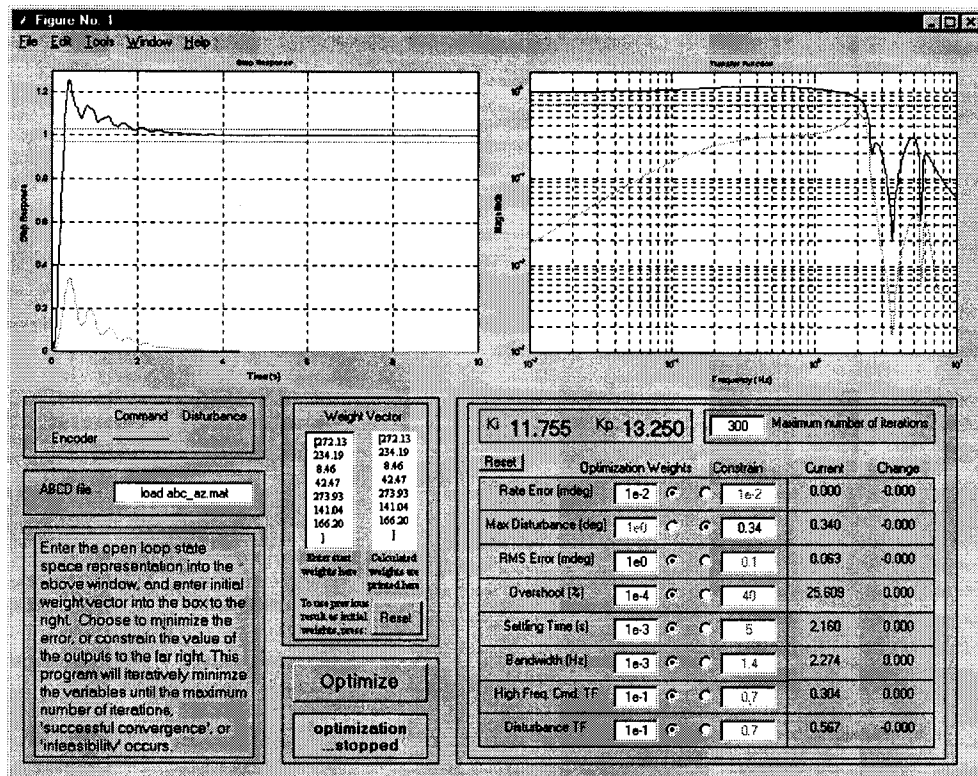


Figure 15c.

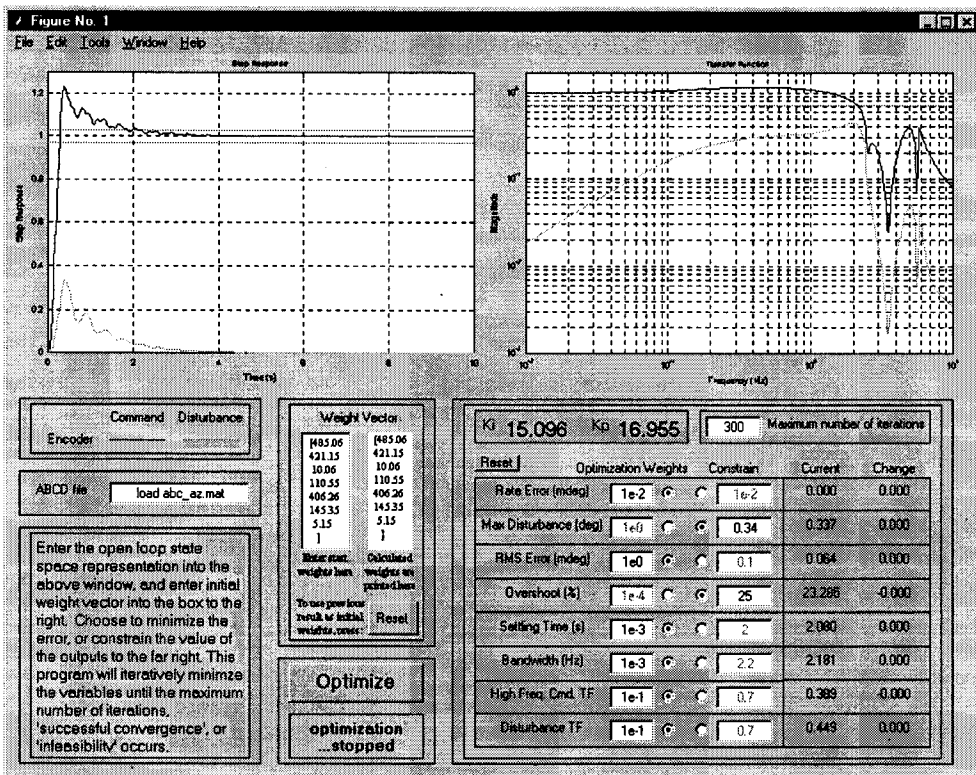


Figure 15d.

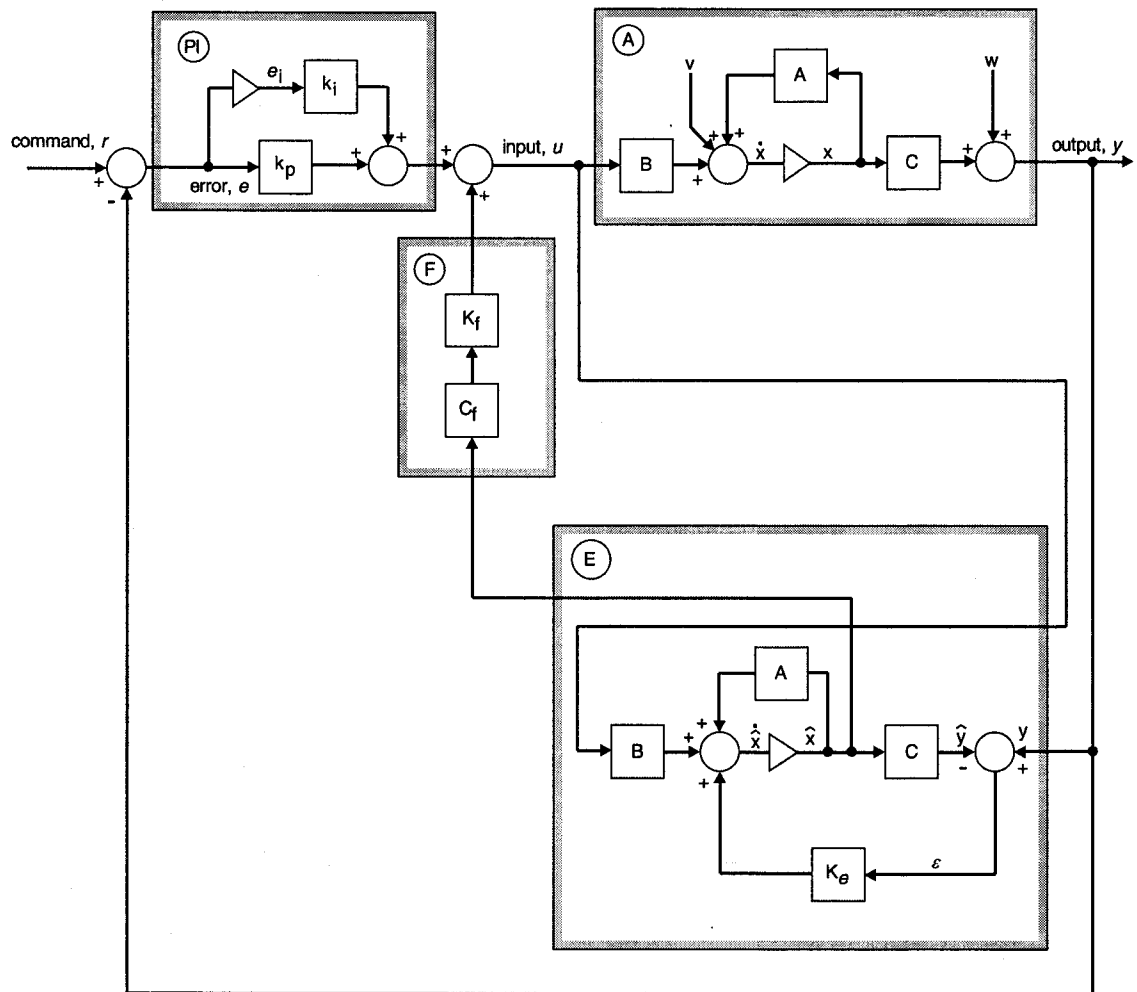


Figure A1.